

ARCHITETTURA DEI SISTEMI ELETTRONICI

LEZIONE N° 9

- **Fenomeni transitori**
- **Somma e differenza di due numeri in C2**
- **Half Adder, Full Adder**
- **Sommatori e Sottrattori di due word di n bit**
- **Livelli di logica**
- **Sommatori veloci**

A.S.E.

9.1

Richiami

- **Realizzazioni diverse della stessa funzione**
- **Mappe di Karnaugh**
- **Implicanti**
- **Implicanti principali**
- **Concetto di minimizzazione (funzione costo)**
- **Sintesi ottima**

A.S.E.

9.2

Tecniche strutturate

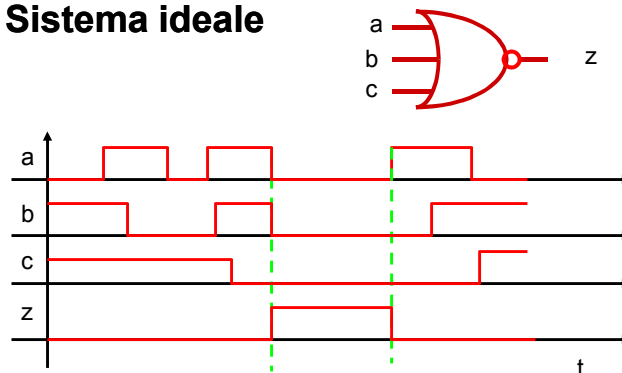
- Il procedimento di sintesi per “ispezione visiva” si può utilizzare fino a 4 ÷ 5 variabili
- Il procedimento di sintesi per “ispezione visiva” può essere anche descritto come processo formale strutturato
- Metodo di Quine McCluskey
- Può essere tradotto in un programma
- La complessità del programma cresce in modo esponenziale con l'aumentare delle variabili
- I programmi attuali usano tecniche euristiche

A.S.E.

9.3

Transitori 1

- Sistema ideale



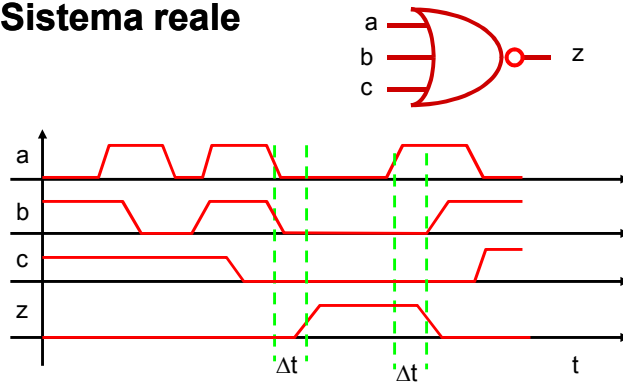
**Le uscite commutano istantaneamente
Nessun ritardo fra ingresso e uscita**

A.S.E.

9.4

Transitori 2

- **Sistema reale**



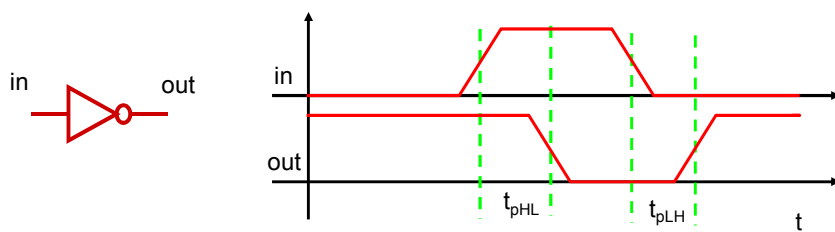
Le uscite commutano in ritardo

A.S.E.

9.5

Ritardo di propagazione

- t_{pHL} e t_{pLH}



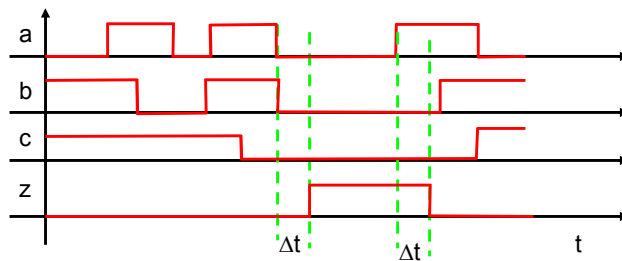
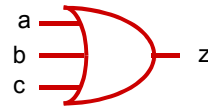
$$t_{pd} = \frac{t_{pHL} + t_{pLH}}{2}$$

A.S.E.

9.6

Transitori 3

- Sistema reale stilizzato



Le forme d'onda sono ideali
Si conservano i ritardi

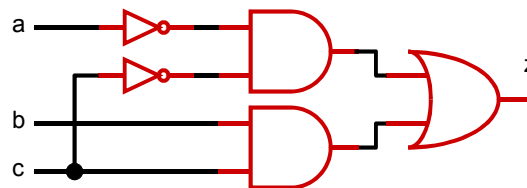
A.S.E.

9.7

Transizioni multiple su gli ingressi

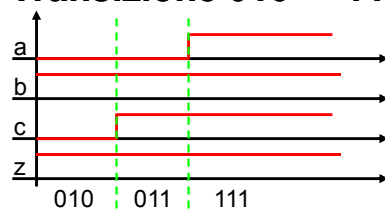
- Possono dare luogo a *glitch*

	bc			
	00	01	11	10
a	0	1	1	1
	0	1	1	1
1				



$$z = \overline{a}c + bc$$

- Transizione 010 → 111



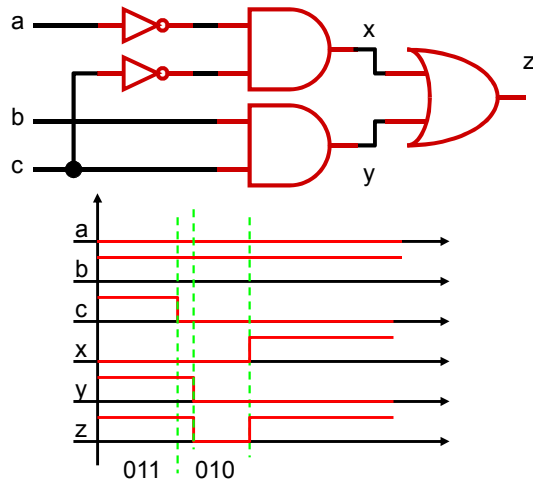
A.S.E.

9.8

Alee Statiche

- Transizione 011 → 010

$a \backslash bc$	00	01	11	10
0	1		1	1
1			1	



- Alea statica di "1"

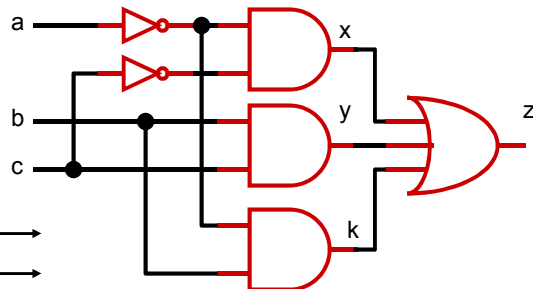
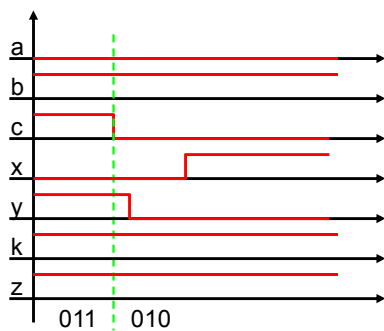
A.S.E.

9.9

Correzione

- Aggiungere implicant per coprire gli "1" adiacenti

$a \backslash bc$	00	01	11	10
0	1		1	1
1			1	



A.S.E.

9.10

Aritmetica binaria 1

- **Somma di due bit**

- $x + y$
- s = Somma
- c = Carry (RIPORTO)

x	y	s	c
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

- **Esempio**

carry	1	1	1				1
	1	0	1	1	0	0	1
	1	1	1	0	1	0	1
	1	1	0	0	1	1	0

$89 + 117 = 206$

A.S.E.

9.11

Aritmetica binaria 2

- **Sottrazione di due bit**

- $x - y$
- d = Differenza
- b = Borrow (Prestito)

x	y	d	b
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

x	y	s	c
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

- **Esempio**

borrow	1	1	1				1
	1	1	0	0	1	1	1
	1	1	1	0	1	0	1
	1	0	1	1	0	0	1

$206 - 117 = 89$

A.S.E.

9.12

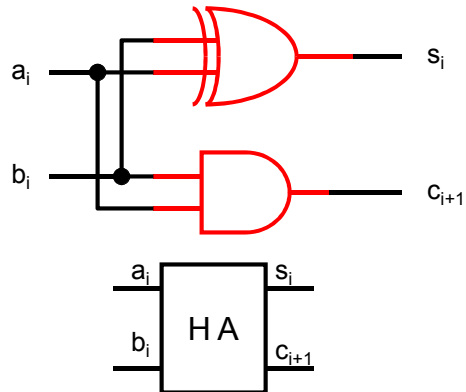
Half Adder

- Somma di due bit

a_i	b_i	s_i	c_{i+1}
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$s_i = a_i \oplus b_i$$

$$c_{i+1} = a_i \cdot b_i$$



A.S.E.

9.13

Full Adder 1

- Somma di due bit compreso il Carry

c_i	a_i	b_i	s_i	c_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

		a_i, b_i			
c_i		00	01	11	10
	s_i	0	1		1
		1	1	1	

		a_i, b_i			
c_i		00	01	11	10
				1	
	c_{i+1}		1	1	1
		1	1	1	1

A.S.E.

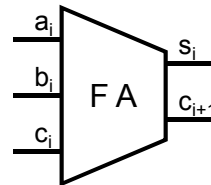
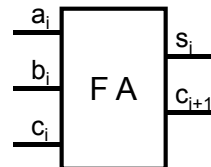
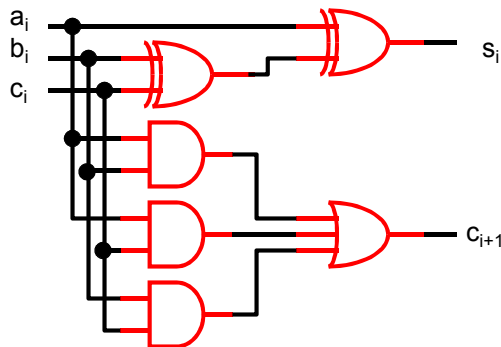
9.14

Full Adder 2

- Lo schema risulta

$$s_i = a_i \oplus b_i \oplus c_i = a_i \oplus (b_i \oplus c_i)$$

$$c_{i+1} = a_i \cdot b_i + a_i \cdot c_i + b_i \cdot c_i$$



A.S.E.

9.15

Full Adder 3

-

c_i	a_i	b_i	s_i	c_{i+1}	$a_i b_i$	$a_i \oplus b_i$	$(a_i \oplus b_i) c_i$	$(a_i \oplus b_i) c_i + a_i b_i$
0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	0	0
0	1	0	1	0	0	1	0	0
0	1	1	0	1	1	0	0	1
1	0	0	1	0	0	0	0	0
1	0	1	0	1	0	1	1	1
1	1	0	0	1	0	1	1	1
1	1	1	1	1	1	0	1	1

A.S.E.

9.16

Full Adder 4

- Somma di due bit compreso il Carry

c_i	a_i	b_i	s_i	c_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$s_i = a_i \oplus b_i \oplus c_i = (a_i \oplus b_i) \oplus c_i$$

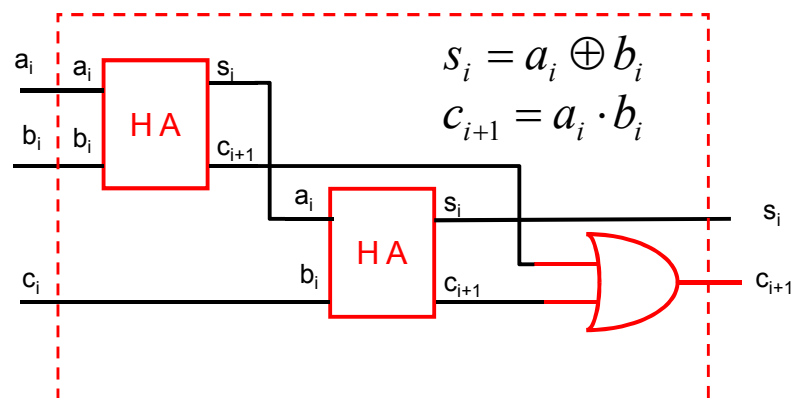
$$\begin{aligned} c_{i+1} &= a_i \cdot b_i \cdot \bar{c}_i + \bar{a}_i \cdot b_i \cdot c_i + a_i \cdot \bar{b}_i \cdot c_i + a_i \cdot b_i \cdot c_i = \\ &= c_i \cdot a_i \cdot \bar{b}_i + c_i \cdot \bar{a}_i \cdot b_i + a_i \cdot b_i \cdot \bar{c}_i + a_i \cdot b_i \cdot c_i = \\ &= c_i \cdot (a_i \cdot \bar{b}_i + \bar{a}_i \cdot b_i) + a_i \cdot b_i \cdot (\bar{c}_i + c_i) = \\ &= c_i \cdot (a_i \oplus b_i) + a_i \cdot b_i \end{aligned}$$

A.S.E.

11.17

Full Adder 5

- Full Adder realizzato con due Half Adder



A.S.E.

9.18

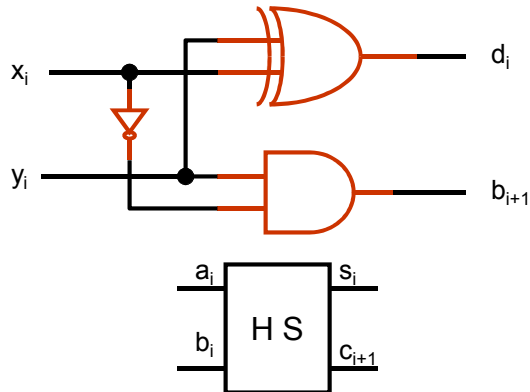
Half Subtractor

- Differenza fra due bit ($x - y$)

x_i	y_i	d_i	b_{i+1}
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$d_i = x_i \oplus y_i$$

$$b_{i+1} = \overline{x_i} \cdot y_i$$



A.S.E.

9.19

Full Subtractor 1

- Differenza fra due bit compreso il Borrow ($x - y$)

b_i	x_i	y_i	d_i	b_{i+1}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

		x_i, y_i			
b_i	d_i	00	01	11	10
		0	1		1
		1	1	1	

		x_i, y_i			
b_i	b_{i+1}	00	01	11	10
		0	1		
		1	1	1	

A.S.E.

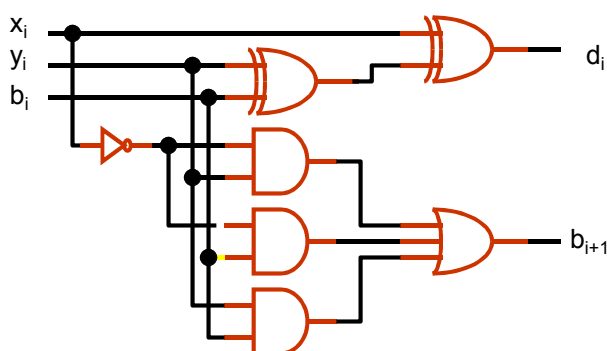
9.20

Full Subtractor 2

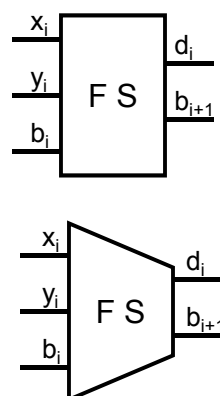
- Lo schema risulta

$$d_i = x_i \oplus y_i \oplus b_i = x_i \oplus (y_i \oplus b_i)$$

$$b_{i+1} = \overline{x_i} \cdot y_i + \overline{x_i} \cdot b_i + y_i \cdot b_i$$



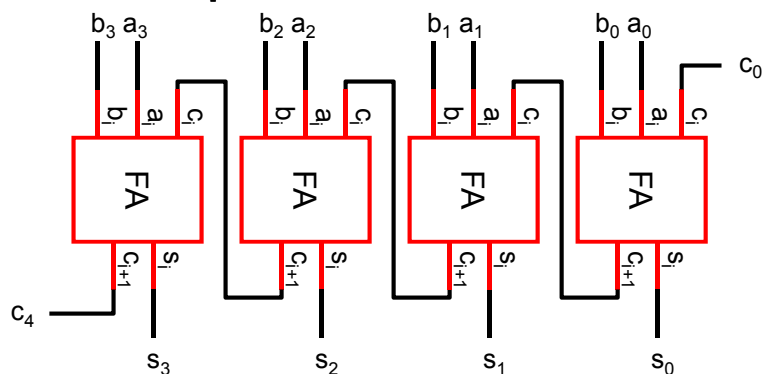
A.S.E.



9.21

Sommatore a riporto seriale (Ripple-Carry Adder)

- Somma di due parole di 4 bit in C. 2



A.S.E.

9.22

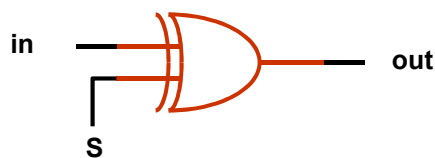
Proprietà dello XOR

- Lo XOR può essere visto come un inverter “programmabile”

S	in	out
0	0	0
0	1	1
1	0	1
1	1	0

per $S = 0$ è $out = in$

per $S = 1$ è $out = \overline{in}$



A.S.E.

9.23

Considerazioni sulla sottrazione

- Si ricorda che

$$W = A - B = A + (-B)$$

- Operando in complemento a 2 si ha

$$-B = (\overline{B}) + 1$$

- Quindi

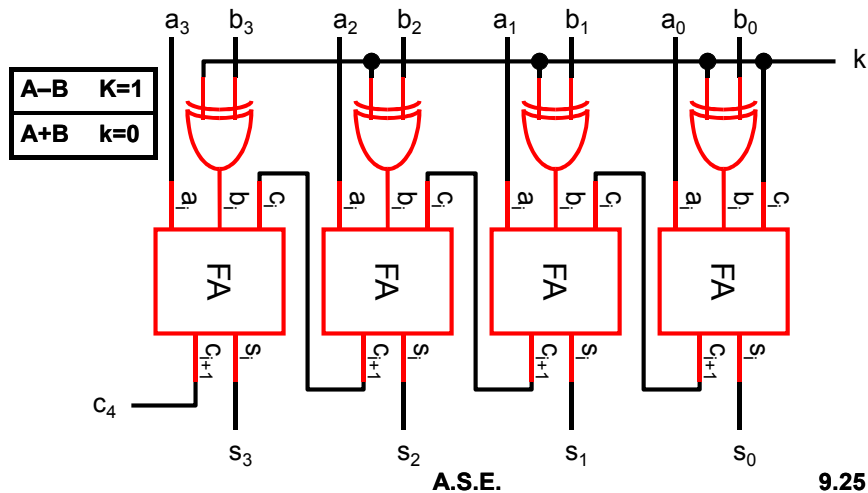
$$W = A - B = A + (\overline{B}) + 1$$

A.S.E.

9.24

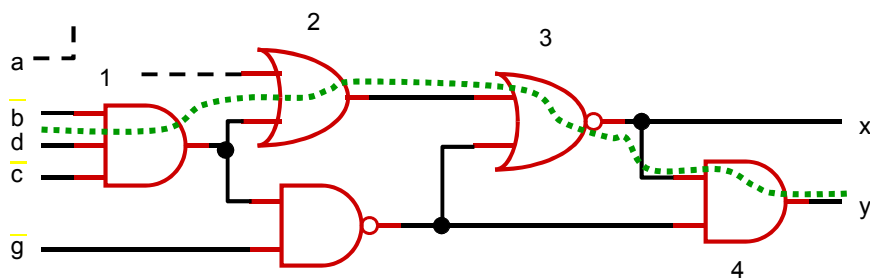
Sommatore/Sottrattore

- In base alle proprietà dello XOR e come si può eseguire la differenza ($A - B$) in C. 2 si ha:



Livelli di logica

- Data una rete combinatoria



- Definizione

- Livelli di logica della rete = numero MAX di blocchi base attraversati passando da un ingresso a una uscita

- NOTA

- La negazione degli ingressi non conta

A.S.E.

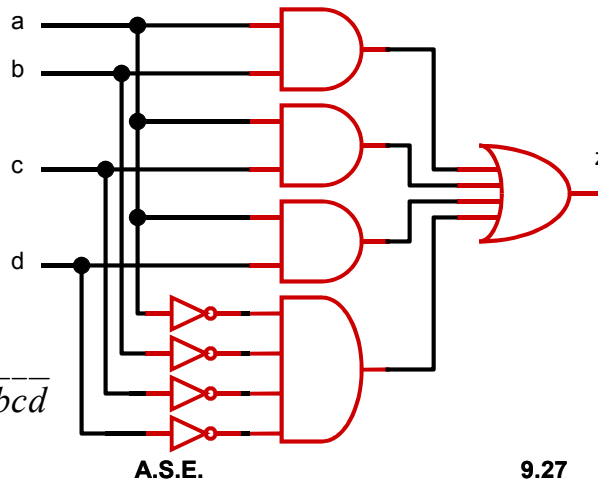
9.26

Sintesi a due livelli

- Le tecniche fin ora viste sono di sintesi a due livelli

ab \ cd	00	01	11	10
00	1			
01				
11	1	1	1	1
10		1	1	1

$$z = ab + ac + ad + \overline{abcd}$$

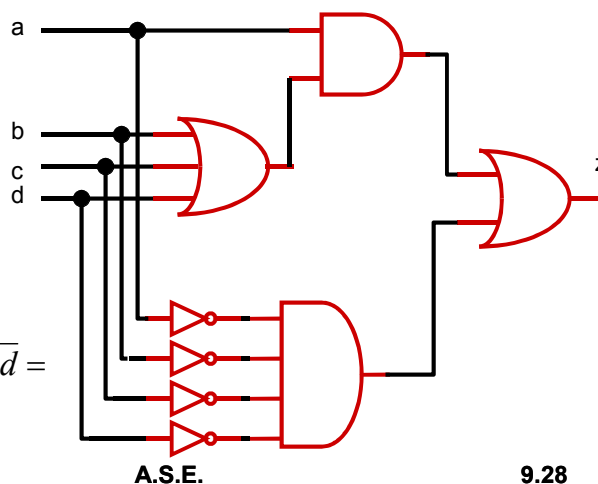


Sintesi a tre livelli

- Si usa un numero inferiore di porte e con meno ingressi

ab \ cd	00	01	11	10
00	1			
01				
11	1	1	1	1
10		1	1	1

$$\begin{aligned} z &= ab + ac + ad + \overline{abcd} = \\ &= a \cdot (b + c + d) + \overline{abcd} \end{aligned}$$



Reti a più uscite

- **Casi visti**
 - più ingressi una uscita
- **Tecniche di minimizzazione viste**
 - Una sola uscita
- **Casi frequenti nella pratica**
 - più ingressi più uscite
- **La minimizzazione delle singole uscite (separatamente) non garantisce la minimizzazione dell'intera rete**
- **Il procedimento di minimizzazione globale risulta molto complesso**

A.S.E.

9.29

Esempio

- **Rete a due uscite**

z

	<i>cd</i>	00	01	11	10
<i>a</i>					
0			1		
1			1	1	

$$z = \bar{c}d + ad$$

w

	<i>cd</i>	00	01	11	10
<i>a</i>					
0		1	1		
1		1			

$$w = \bar{c}\bar{d} + \bar{a}c$$

$$z = \bar{a}cd + ad$$

$$w = \bar{a}cd + \bar{c}\bar{d}$$

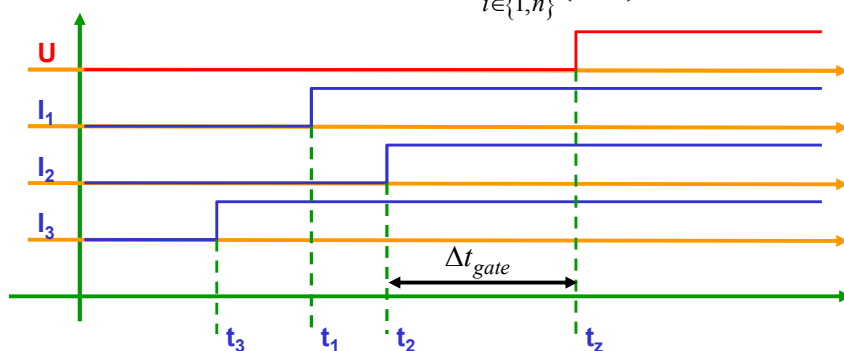
A.S.E.

9.30

Tempo di ritardo

- Per una porta logica si ha

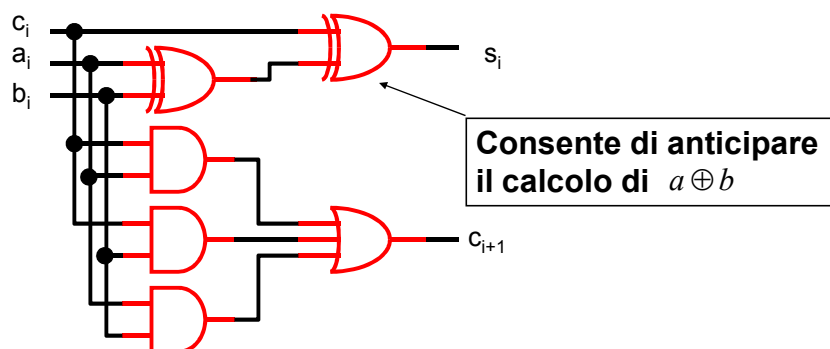
$$t_{gate} = \Delta T_{gate} + \max_{i \in \{1, n\}} \{t_{in, i}\}$$



A.S.E.

9.31

Ritardi del FULL ADDER 1



$$t_{s_i} = \Delta T_{xor, 2} + \max \{c_i, t_{a_i \oplus b_i}\}$$

$$t_{c_{i+1}} = \Delta T_{or, 3} + \max \{t_{(a_i \square b_i)}, t_{(a_i \square c_i)}, t_{(b_i \square c_i)}\}$$

A.S.E.

9.32

Ritardi del FULL ADDER 2

- Per il C_{i+1} si ha

$$t_{c_{i+1}} = \Delta T_{or,3} + \max \left\{ \begin{array}{l} \Delta T_{and,2} + \max \{t_{a_i}, t_{b_i}\}, \\ \Delta T_{and,2} + \max \{t_{a_i}, t_{c_i}\}, \\ \Delta T_{and,2} + \max \{t_{b_i}, t_{c_i}\} \end{array} \right\} =$$

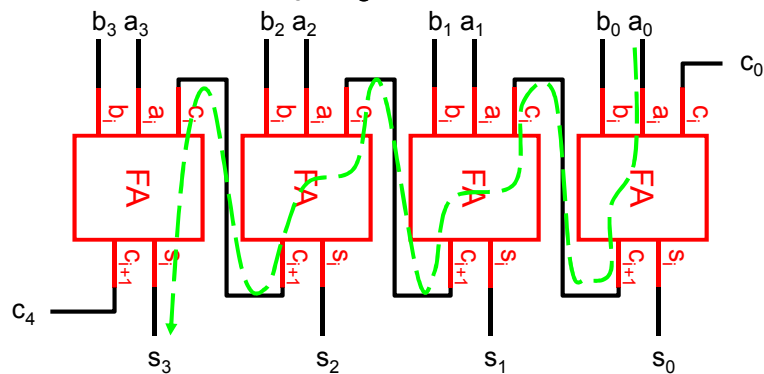
$$= \Delta T_{or,3} + \max \left\{ \Delta T_{and,2} + \max \{t_{a_i}, t_{b_i}, t_{c_i}\} \right\}$$

A.S.E.

11.33

Tempo di ritardo nel Sommatore

- T_c = ritardo del Carry, T_s = ritardo della somma



$$t_{s_{n-1}} = \Delta T_{xor2} + \max \{t_{c_{n-1}}, t_{a_{n-1}} \oplus t_{b_{n-1}}\} = \Delta T_{xor2} + t_{c_{n-1}}$$

A.S.E.

9.34

Ritardo del sommatore Ripple Carry

- Per il Carry iesimo si ha:

$$t_{c_{i+1}} = \Delta T_{or,3} + \max \left\{ \Delta T_{and,2} + \max \{t_{a_i}, t_{b_i}, t_{c_i}\} \right\}$$

$$= \Delta T_{or,3} + \Delta T_{and,2} + t_{c_i}$$

- Per il Full Adder si ha:

$$\Delta T_c = \Delta T_{or,3} + \Delta T_{and,2}$$

- Quindi

Ritardo del sommatore Ripple Carry

$$t_{c_{n-1}} = (n-1) \Delta T_c$$

$$t_{s_{n-1}} = \Delta T_{xor,2} + (n-1) \Delta T_c$$

A.S.E.

12.35

Sommatori veloci

- Considerazioni sul Carry

c_i	a_i	b_i	s_i	c_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$s_i = a_i \oplus b_i \oplus c_i$$

$$c_{i+1} = a_i \cdot b_i \cdot \overline{c_i} + \overline{a_i} \cdot b_i \cdot c_i + a_i \cdot \overline{b_i} \cdot c_i + a_i \cdot b_i \cdot c_i =$$

$$= (a_i \cdot b_i \cdot \overline{c_i} + a_i \cdot b_i \cdot c_i) + (\overline{a_i} \cdot b_i \cdot c_i + a_i \cdot \overline{b_i} \cdot c_i)$$

$$c_{i+1} = a_i \cdot b_i + c_i \cdot (a_i \oplus b_i)$$

$$G_i = a_i \cdot b_i$$

$$P_i = a_i \oplus b_i$$

$$c_{i+1} = G_i + c_i \cdot P_i$$

$$S_i = P_i \oplus c_i$$

A.S.E.

9.36

Carry Look-Ahead Adder

- Quindi risulta

$$* C_1 = G_0 + P_0 C_0$$

$$* C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$* C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

$$* C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$$

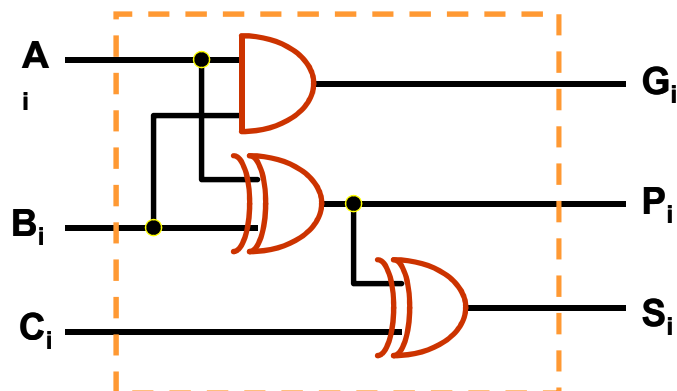
– I vari Carry possono essere generati simultaneamente

A.S.E.

9.37

Blocco base

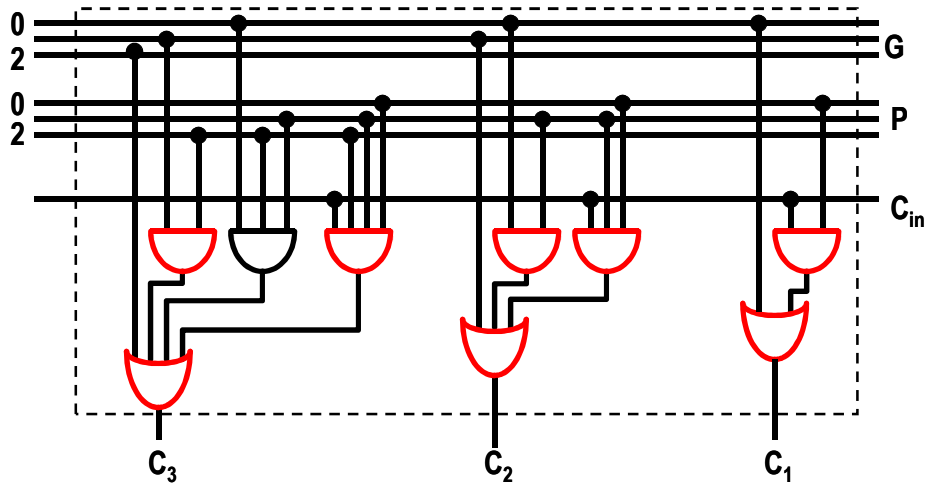
$$\begin{aligned} G_i &= A_i B_i & P_i &= A_i \oplus B_i \\ S_i &= P_i \oplus C_i \end{aligned}$$



A.S.E.

9.38

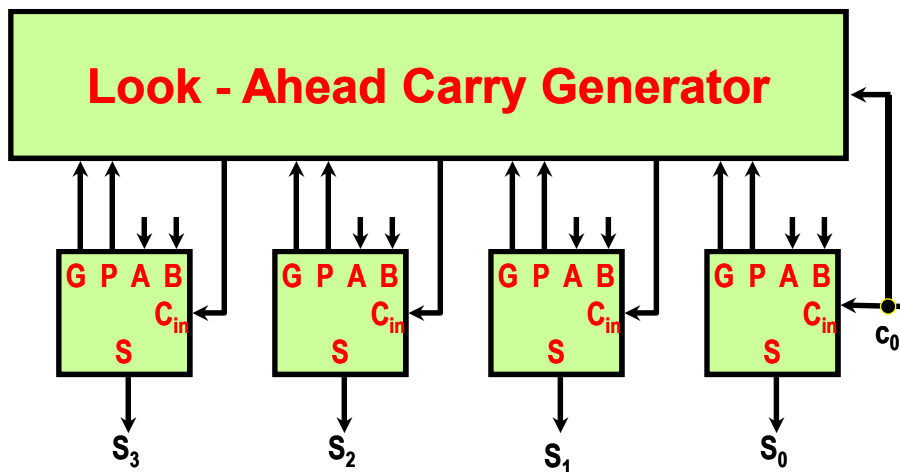
Look - Ahead Carry Generator Schema



A.S.E.

9.39

Schema del sommatore



A.S.E.

9.40

Conclusioni

- **Fenomeni transitori**
 - Commutazioni multiple degli ingressi
 - Alee statiche
- **Somma e differenza di due numeri in C2**
- **Half Adder**
- **Full Adder**
- **Sommatori e Sottrattori di due word di n bit**
- **Sommatori veloci**

A.S.E.

9.41